

[](https://www.tutorialr.com/tutorials/)

Windows App SDK

Info Bar





# Info Bar

**Info Bar** shows how you can use the **InfoBar** using the **Windows App SDK** which is a **Control** that can be

used to display status messages with different levels of **Severity** in an application.

## Step 1

Follow **Setup and Start** on how to get **Setup** and **Install** what you need for **Visual Studio 2022** and **Windows App SDK**.

|  |  |
| --- | --- |
| In **Windows 11** choose **Start** and then find or search for **Visual Studio 2022** and then select it. | Text  Description automatically generated |
| Once **Visual Studio 2022** has started select **Create a new project**. | **Graphical user interface, text  Description automatically generated** |
| Then choose the **Blank App, Packages (WinUI in Desktop)** and then select **Next**. | **Graphical user interface, text  Description automatically generated** |
| After that in **Configure your new project** type in the **Project name** as *InfoBar,* then select a Location and then select **Create** to start a new **Solution**. | **Graphical user interface, text, application, email  Description automatically generated** |

## Step 2

|  |  |
| --- | --- |
| Within **Solution** **Explorer** for the **Solution** and double-click on **MainWindow.xaml** to see the **XAML** for the **Main Window**. |  |

## Step 3

In the **XAML** for **MainWindow.xaml** there be some **XAML** for a **StackPanel**, this should be **Removed** by removing the following:

<StackPanel Orientation="Horizontal"

HorizontalAlignment="Center" VerticalAlignment="Center">

<Button x:Name="myButton" Click="myButton\_Click">Click Me</Button>

</StackPanel>

## Step 4

While still in the **XAML** for **MainWindow.xaml** above **</Window>**, type in the following **XAML**:

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<ComboBox Grid.Row="0" Margin="25" Name="Options"

HorizontalAlignment="Stretch" SelectionChanged="Options\_SelectionChanged">

<ComboBoxItem IsSelected="True">Informational</ComboBoxItem>

<ComboBoxItem>Success</ComboBoxItem>

<ComboBoxItem>Warning</ComboBoxItem>

<ComboBoxItem>Error</ComboBoxItem>

</ComboBox>

<InfoBar Margin="50" Grid.Row="1" Name="Display"

Severity="Informational" IsClosable="False" IsOpen="True"

Title="Informational" Message="Hello World" />

</Grid>

This **XAML** features a **Grid** with two rows, denoted with **RowDefinition**, the **Height** of **Auto** will accommodate the **ComboBox** or drop-down list which has **ComboBoxItems** the **Severity** level, when an option is selected this will trigger the **Event** of **SelectionChanged** and this will Invokea **Method** of **Options\_SelectionChanged**. Then there is a **RowDefinition** with the **Height** of **\*** which will be other part of the **Grid** for the **InfoBar**. It has **Severity** set to **Informational** along with the **Title**, the **Property** for **IsClosable** is **False** which means that it cannot be closed, if this was set to **True** then it could be closed. The **Property** for **IsOpen** controls if the **InfoBar** is displayed or not. There is also a **Property** for **Message** which will be what will be displayed to the user in the **InfoBar**.

## Step 5

|  |  |
| --- | --- |
| Then, within **Solution** **Explorer** for the **Solution** select the arrow next to **MainWindow.xaml** then double-click on **MainWindow.xaml.cs** to see the **Code** for the **Main Window**. |  |

## Step 6

In the **Code** for **MainWindow.xaml.cs** there be a **Method** of **myButton\_Click(...)** this should be **Removed** by removing the following:

private void myButton\_Click(object sender, RoutedEventArgs e)

{

myButton.Content = "Clicked";

}

## Step 7

Once **myButton\_Click(...)** has been removed, below the end of **public MainWindow() { ... }** type in the following **Code**:

private void Options\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

if (Display != null)

{

string severity = (Options.SelectedItem as ComboBoxItem).Content as string;

Display.Severity = Enum.Parse<InfoBarSeverity>(severity);

Display.Title = severity;

}

}

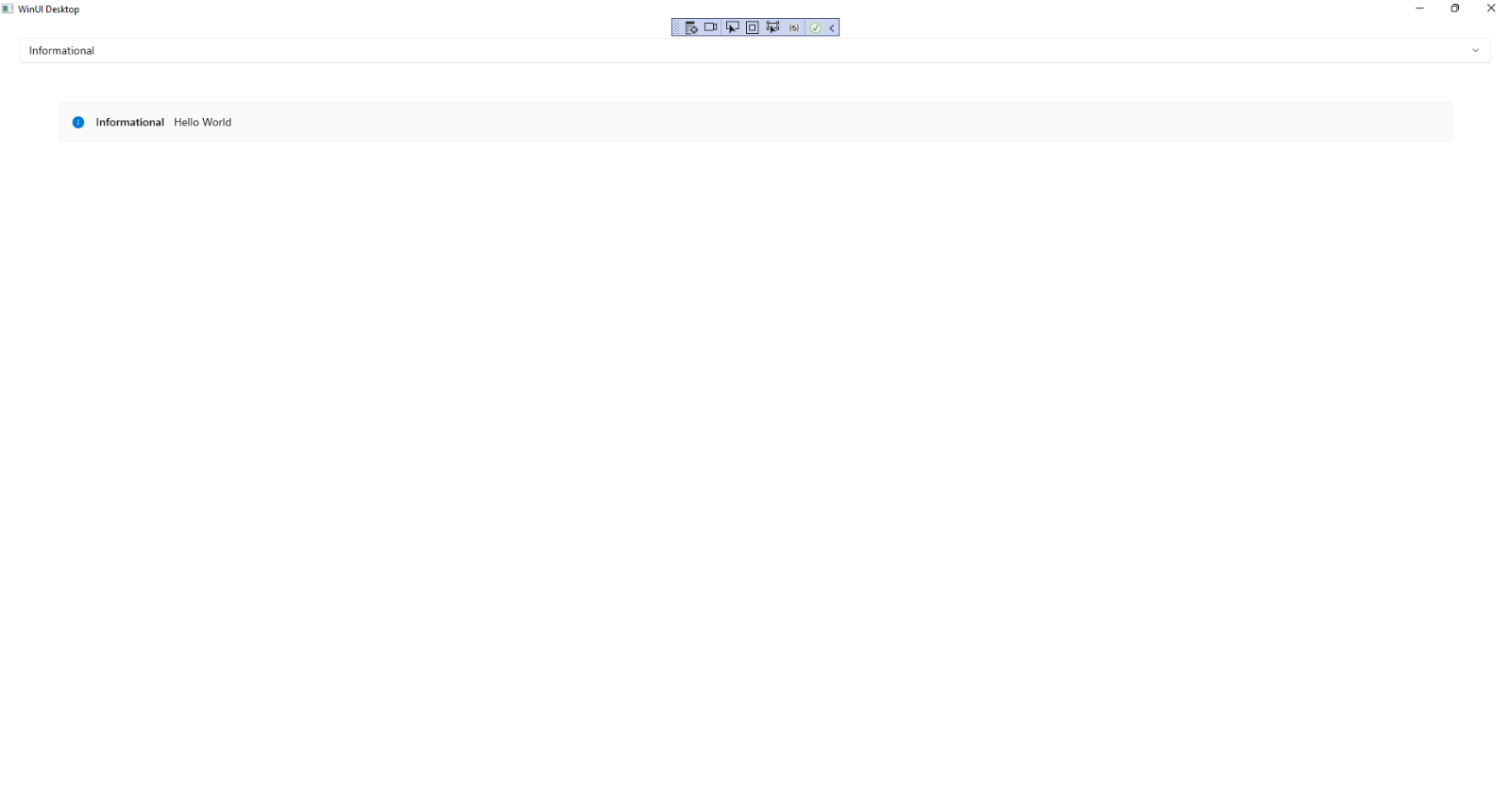
The **Method** of **Options\_SelectionChanged** will be triggered by the **Event** of **SelectionChanged** which is when an item in the **ComboBox** is selected. The first thing is to check that **Display** has a value by seeing if it is not **null** and if it is then the next thing is to get the **Content** of the **SelectedItem** from the **ComboBox**. This is then used to set the **Title** and the **Severity**, which uses the **Method** for **Enum.Parse** to convert this to a value of **InfoBarSeverity**, for the **InfoBar**.

## Step 8

|  |  |
| --- | --- |
| That completes the **Windows App SDK** application. In **Visual Studio 2022** from the **Toolbar** select **InfoBar (Package)** to **Start** the application. |  |

## Step 9

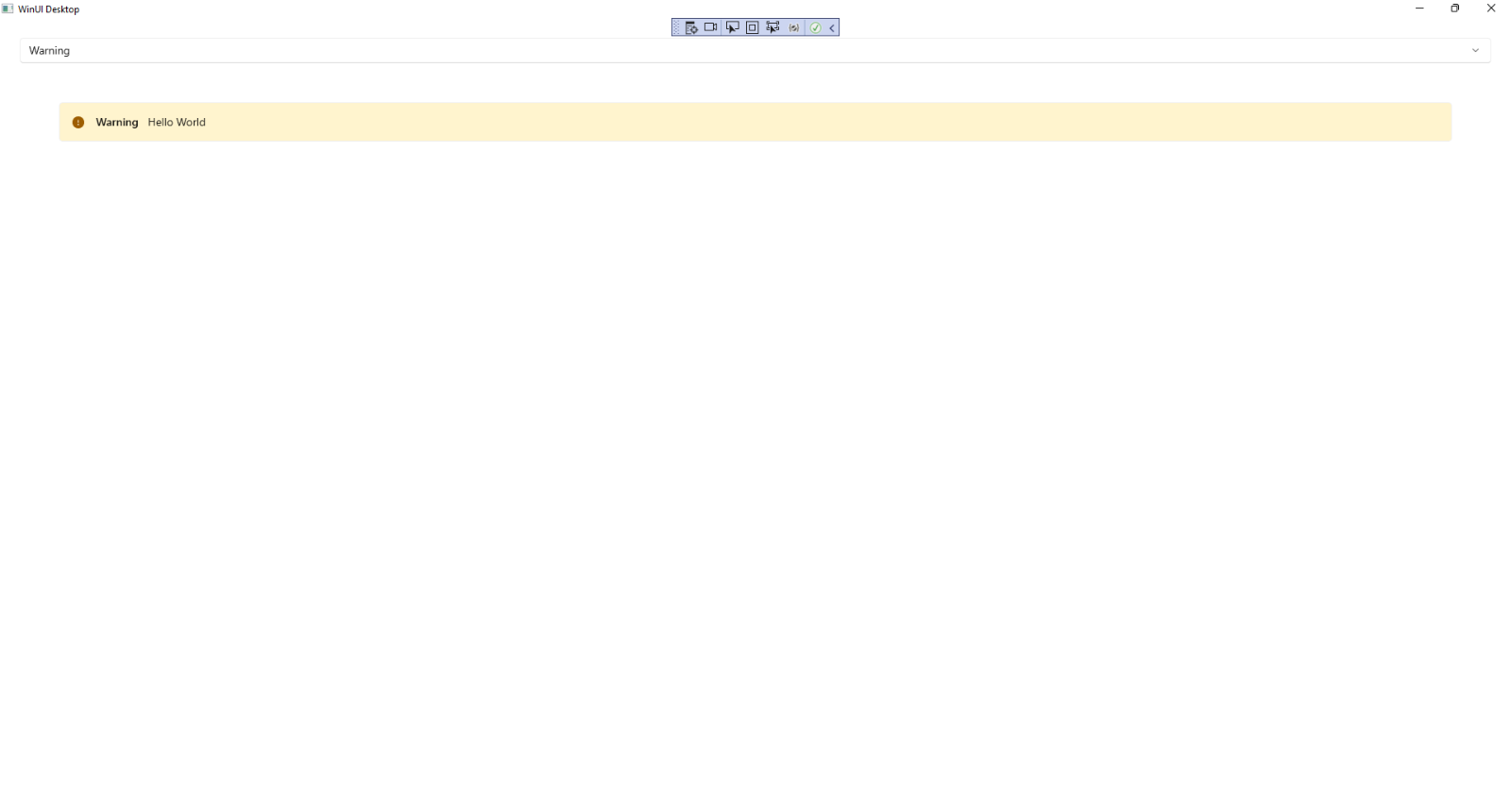
Once running you should see the **InfoBar** with **Severity** using the **InfoBarSeverity** of **Informational**

****

## Step 10

If you **Select** one of the items in the **ComboBox** or drop-down list, the **InfoBar** will then be displayed using

the **InfoBarSeverity** that was selected such as **Warning**.



## Step 11

|  |  |
| --- | --- |
| To **Exit** the **Windows App SDK** application, select the **Close** button from the top right of the application as that concludes this **Tutorial** for **Windows App SDK** from [tutorialr.com](https://tutorialr.com)! |  |